Cryptographie Symétrique Moderne

Partie 1

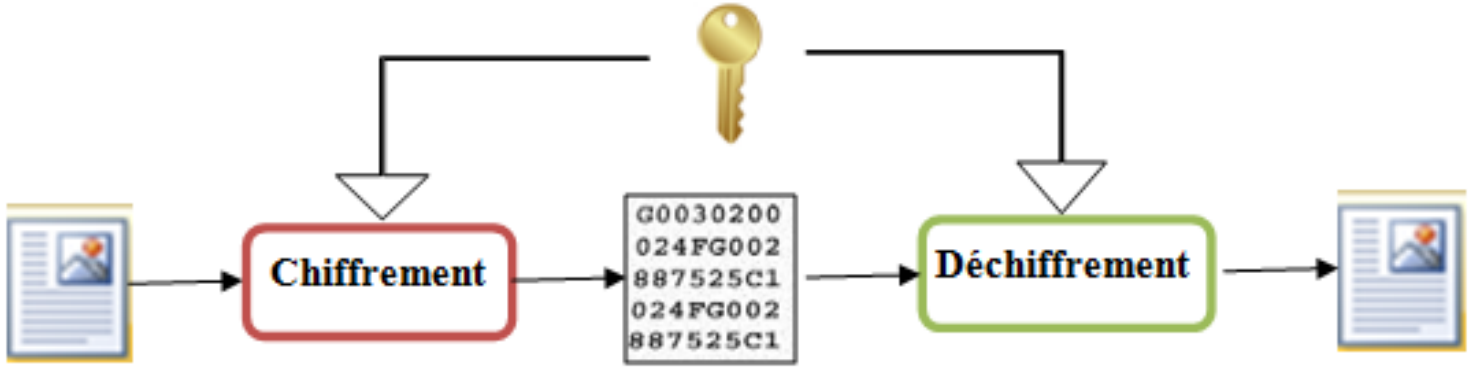
Dr. Noureddine Chikouche
noureddine.chikouche@univ-msila.dz
https://sites.google.com/view/chikouchenoureddine

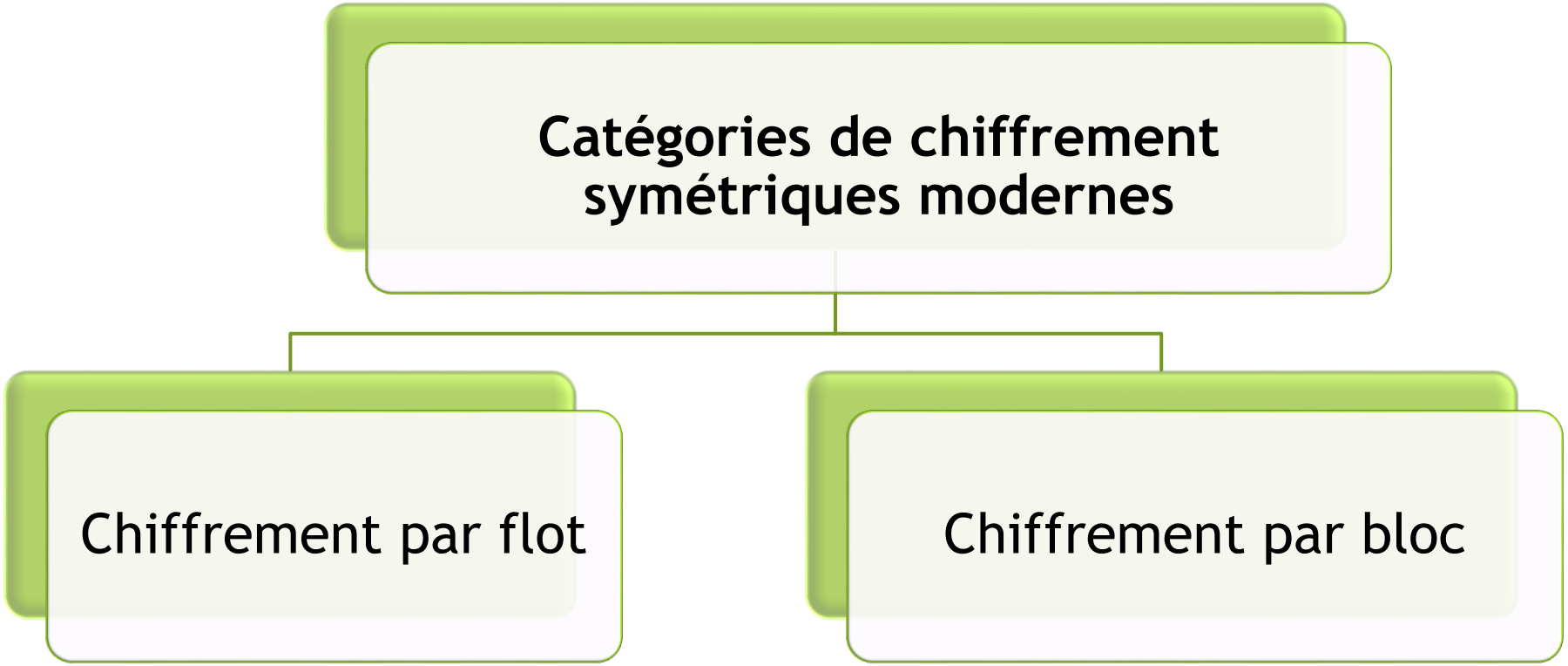
Plan du cours

- Introduction
- Les catégories de chiffrement symétriques modernes
  - Chiffrement par flot
  - Chiffrement par bloc
- DES

Introduction

Cryptographie symétrique




Catégories de chiffrement symétriques modernes
Chiffrement par flot
Chiffrement par bloc

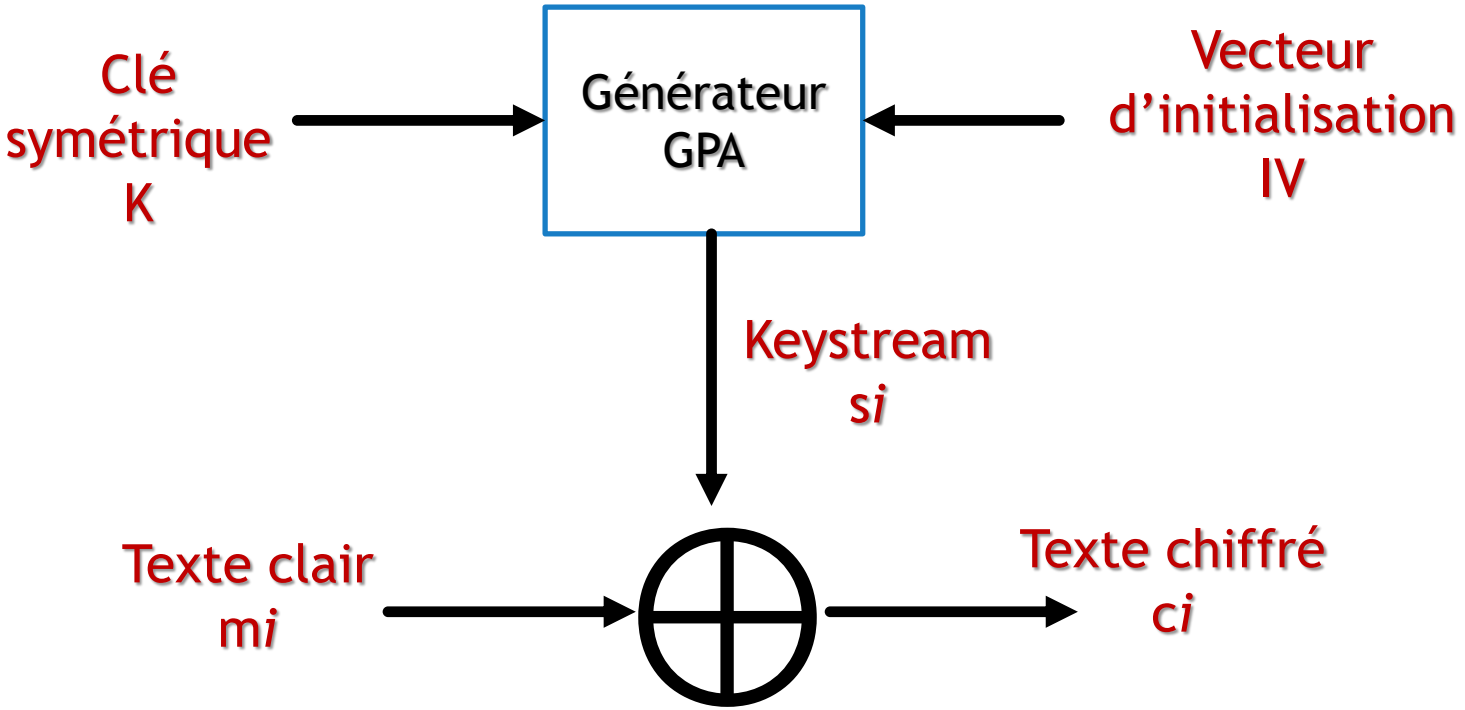
Chiffrement par flot

- Chiffrement de flux / chiffrement par flot / stream cipher
- On chiffre chaque bit/octet du message clair avec une nouvelle clé appelée clé de flot (*keystream*).
- Keystream (*s*) est une séquence des bits générée à partir de la clé symétrique (*K*).

Chiffrement par flot

- Pour générer une keystream, on utilise un générateur pseudo-aléatoire (GPA).
- La taille de la clé générée est égale à celle du texte clair.
- Pour chiffrer un message, on utilise le principe de masque jetable (*One Time Pad*).
- Fonction de chiffrement:
  - $c_i = E_{s_i}(m_i) = m_i \oplus s_i$
- Fonction de déchiffrement:
  - $m_i = D_{s_i}(c_i) = c_i \oplus s_i$

Chiffrement par flot



Chiffrement par Flot: Exemple

- **Exemple:**
- Texte clair (*m*): 01010011 01000001
- Keystream (*s*) = 01110111 01110111
- Chiffrement

01010011 01000001

$\oplus$

01110111 01110111

=

00100100 00110110

Chiffrement par flot

- **Importance du chiffrement par flot:**
  - Il est très rapide.
  - Implémentation matériel et logiciel facile.
  - Il est adaptés aux applications temps réel.
- **Limitation:**
  - Problème de propagation d’erreur (synchronisation).

Chiffrement par flot

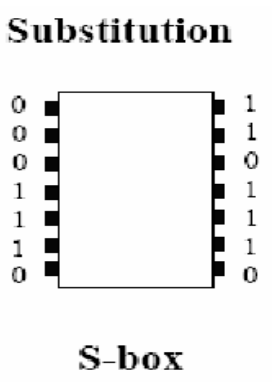
- RC4, utilisé notamment par le protocole WEP pour sécuriser les réseaux sans fils.
- A5/1, utilisé dans les téléphones mobiles de type GSM pour chiffrer la communication par radio entre le mobile et l'antenne-relais la plus proche,
- E0, utilisé par le protocole Bluetooth.
- Autres:

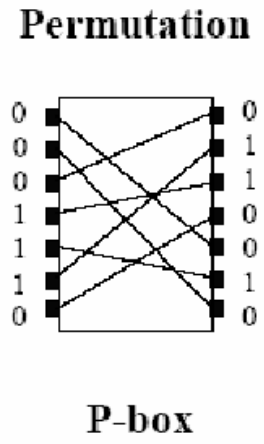
FISH - Helix · ISAAC · LEVIATHAN · MUGI · Panama ·SEAL · SOBER · WAKE

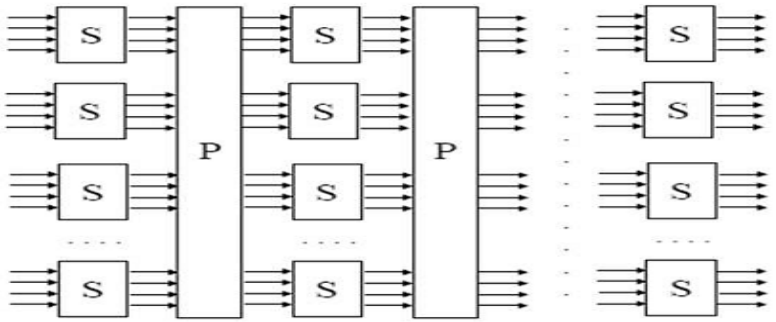
Chiffrement par Bloc

Principe:

1. Le message est découpé en blocs de taille constante,
2. On utilise la clé pour chiffrer chaque bloc.
3. Chaque bloc est chiffré indépendamment de la valeur des autres blocs.
4. Il utilise: chiffrement par substitution (confusion) / transposition (diffusion)/ produit (plus robuste).

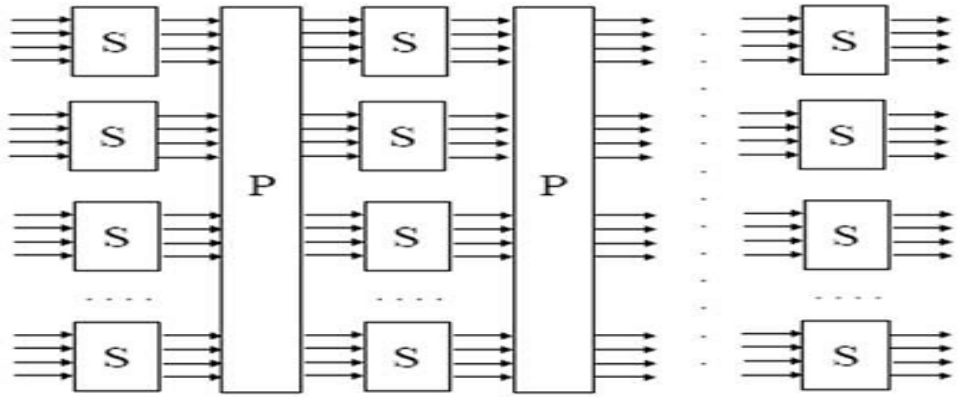






Chiffrement par Bloc

- C. Shannon a prouvé que la combinaison de confusion et diffusion permet d'obtenir une sécurité convenable.
  - Confusion (S-Box): masquer la relation entre le texte clair et le texte chiffré.
  - Diffusion (P-Box): cacher la redondance en répartissant l'influence d'un bit de clé sur tout le chiffré.

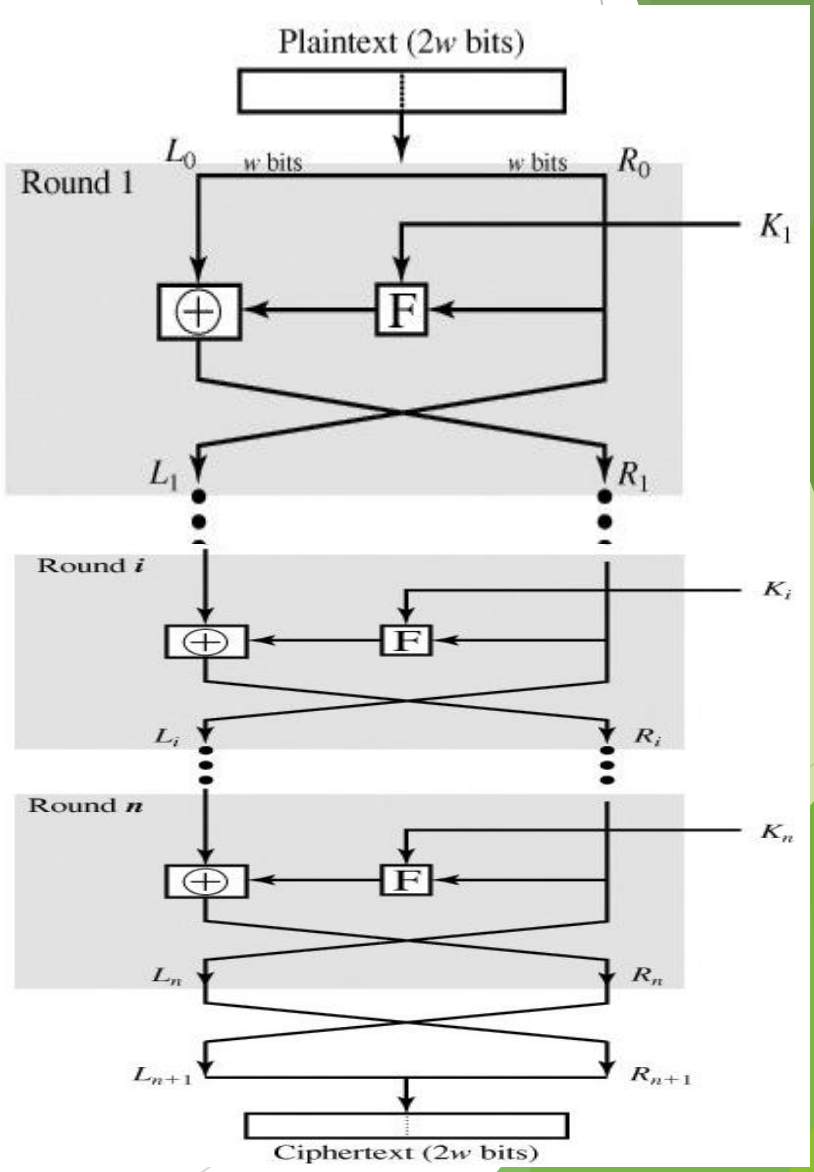


DES (Data Encryption Standard)

Structure de chiffrement de Feistel

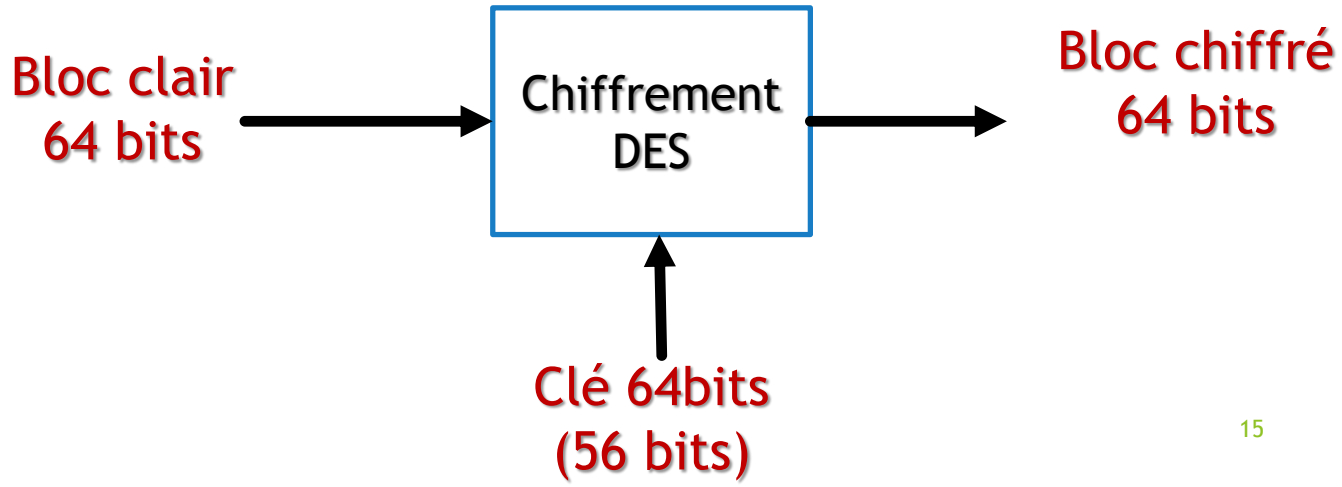
- Inventé par Horst Feistel en 1973 (IBM).
- La longueur de l'entrée est 2w (bits), celle de la clé est K.
- Diviser l'entrée en 2 moitiés L_0 et R_0.
- Pour chaque ronde, on calcule les nouveaux, L_i et R_i.

$$L_i = R_{i-1},$$
$$R_i = L_{i-1} \oplus F_K(R_{i-1})$$

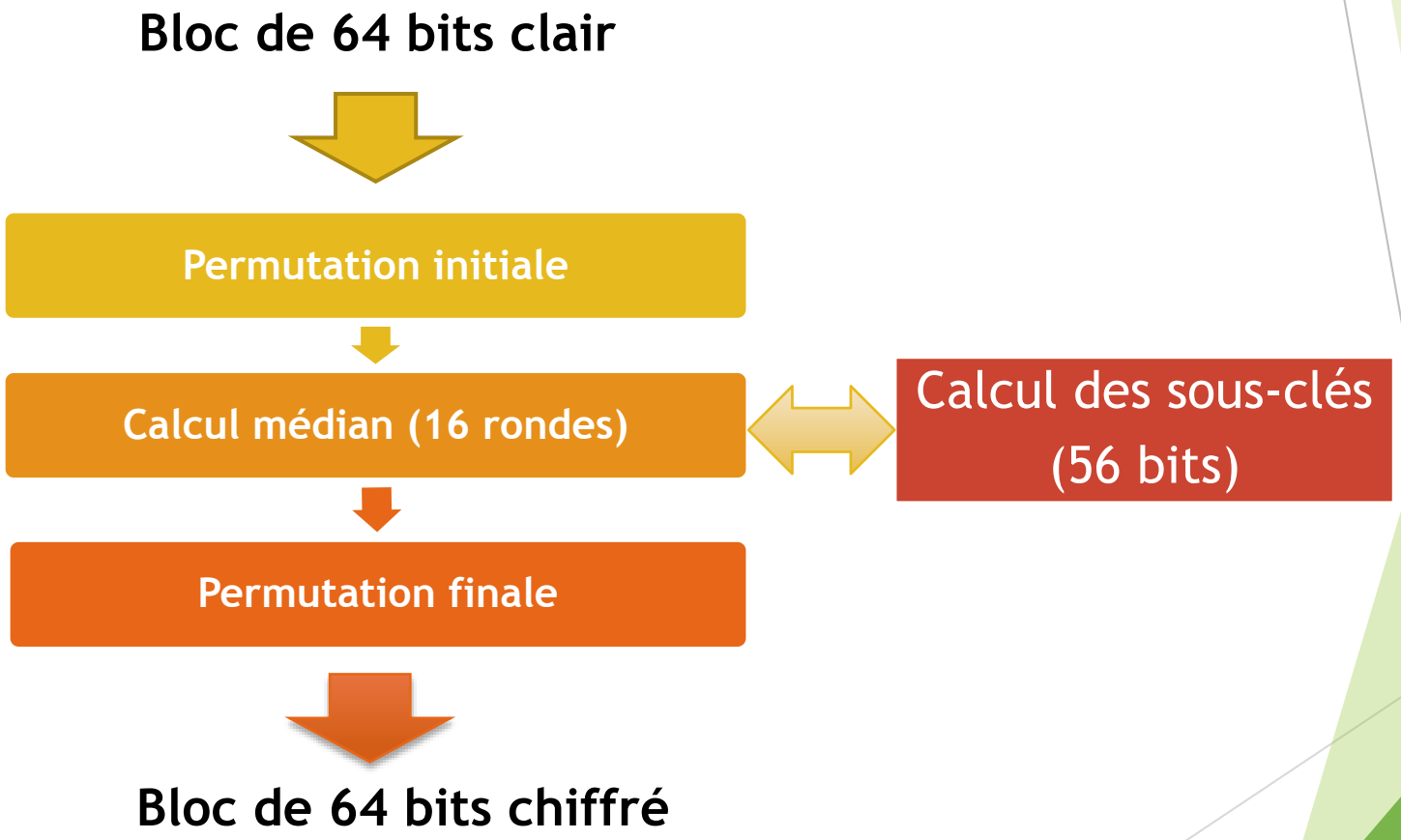


DES (Data Encryption Standard)

- Conçu par IBM en 1973.
- DES adapté comme standard en 1977.
- DES a été l'algorithme officiel de l'administration américaine jusqu'en 1999.
- DES est basée sur le chiffrement de Feistel légèrement modifié avec l'alphabet {0,1} et la longueur des blocs 64.



DES (Data Encryption Standard)



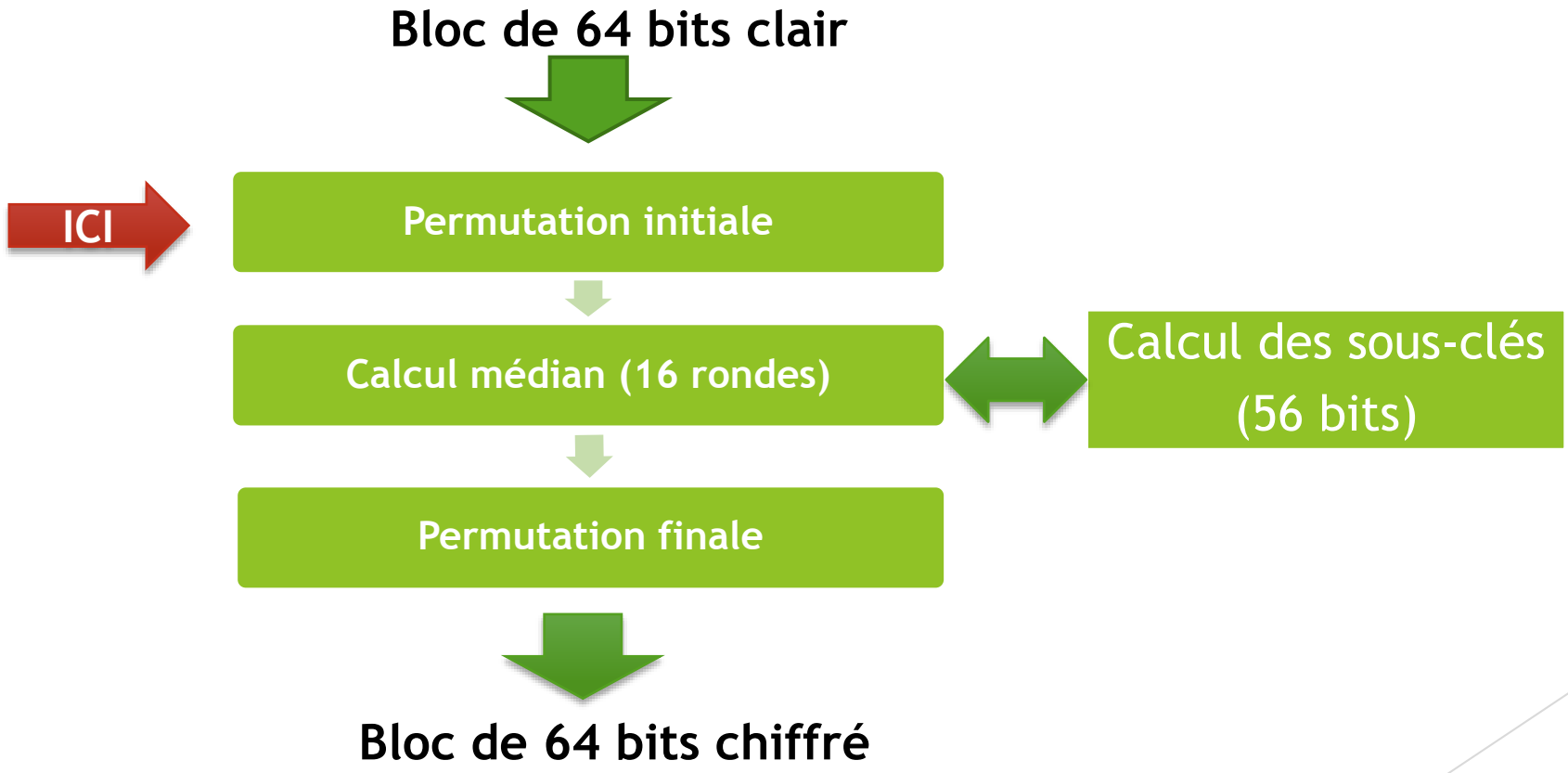
DES (Data Encryption Standard)

- Exemple de bloc:

133457799BBCDFF1

0	0	0	1	0	0	1	1
0	0	1	1	0	1	0	0
0	1	0	1	0	1	1	1
0	1	1	1	1	0	0	1
1	0	0	1	1	0	1	1
1	0	1	1	1	1	0	0
1	1	0	1	1	1	1	1
1	1	1	1	0	0	0	1

DES (Data Encryption Standard)

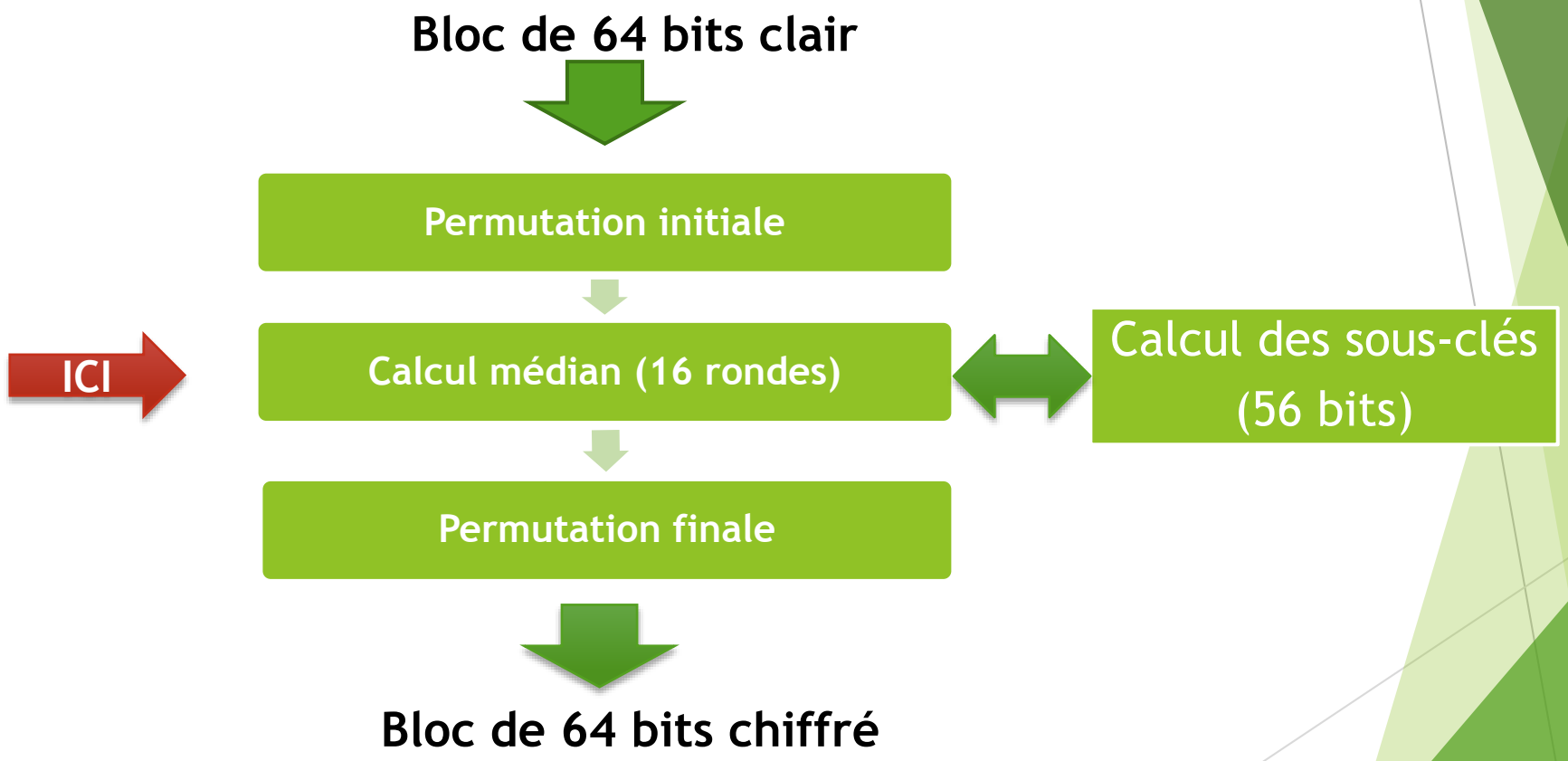


DES: Permutation initiale

- Les 64 bits du texte clair sont soumis à la permutation initiale (PI ou IP) pour produire le texte brouillé selon le tableau suivant:

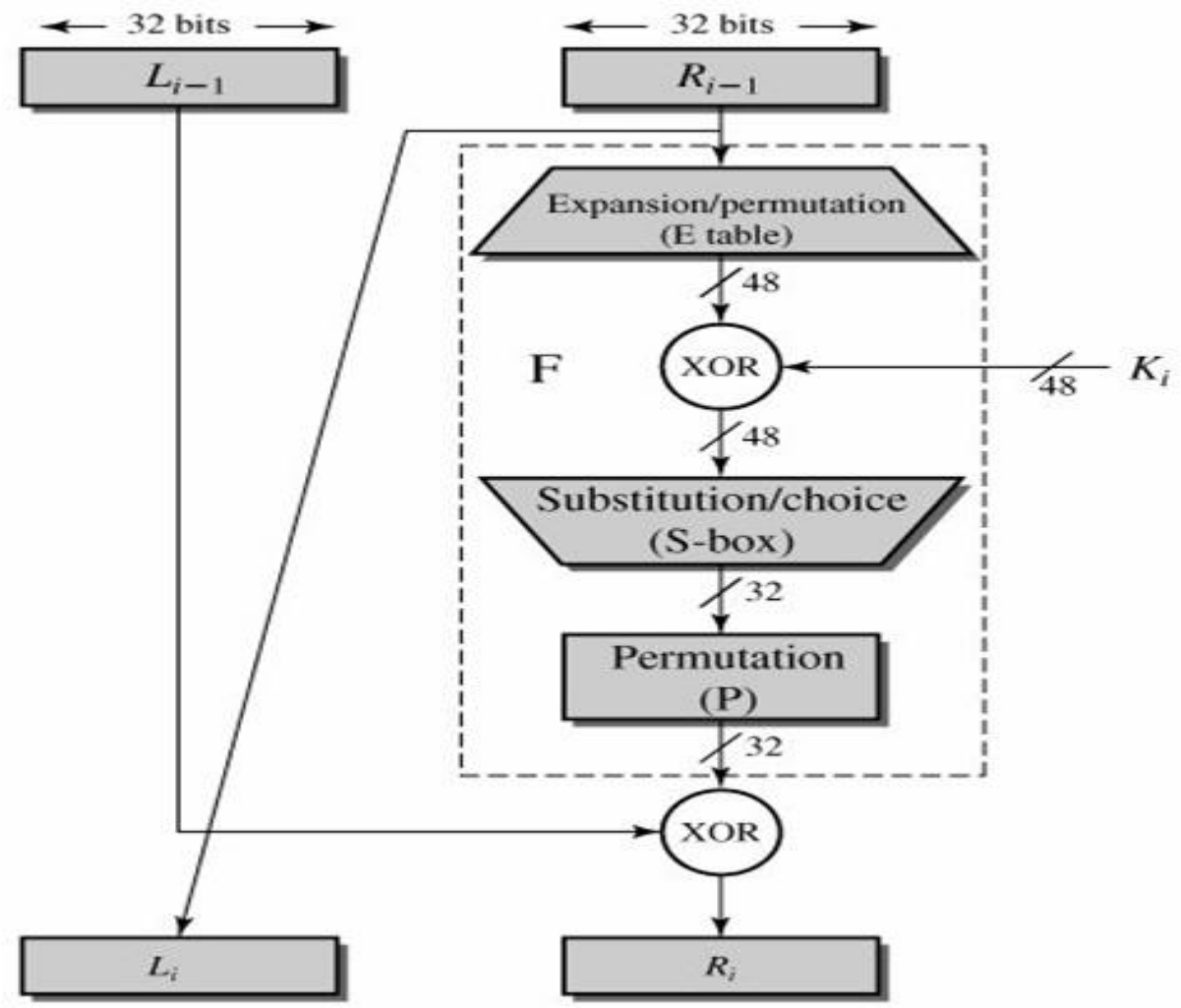
58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

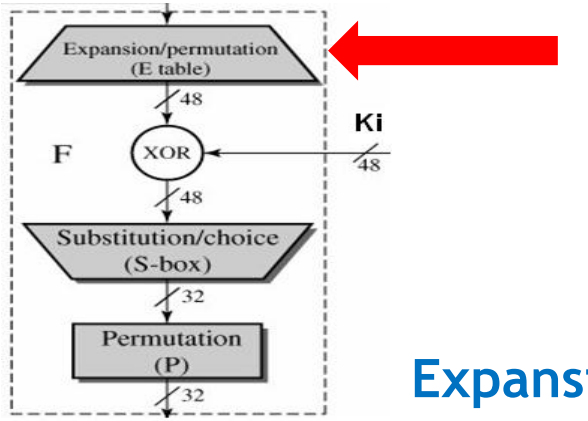
DES (Data Encryption Standard)



DES: Calcul médian

- $Li = R_{i-1}$
- $R_i = L_{i-1} \oplus F(R_{i-1}, K_i)$
- $Ki = G(K,i)$





DES: Calcul médian

Expansion/Permutation (E table):

- Calcul la fonction de développement E:

  32 bits → 48 bits.

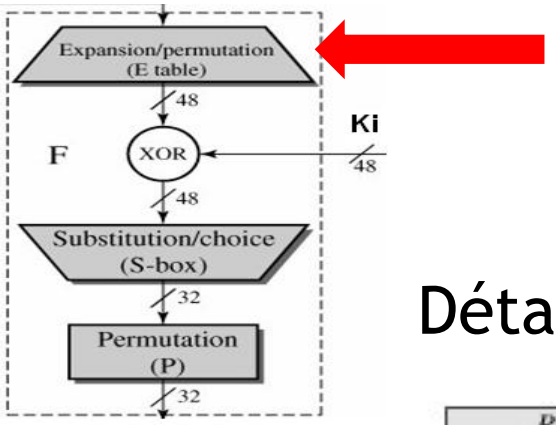
32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

- Certains bits de l'entrée sont dupliqués.

- Ensuite, on calcule $E(R) \oplus K$ et le résultat est découpé en 8 blocs Bi de longueur 6, ce qui donne:

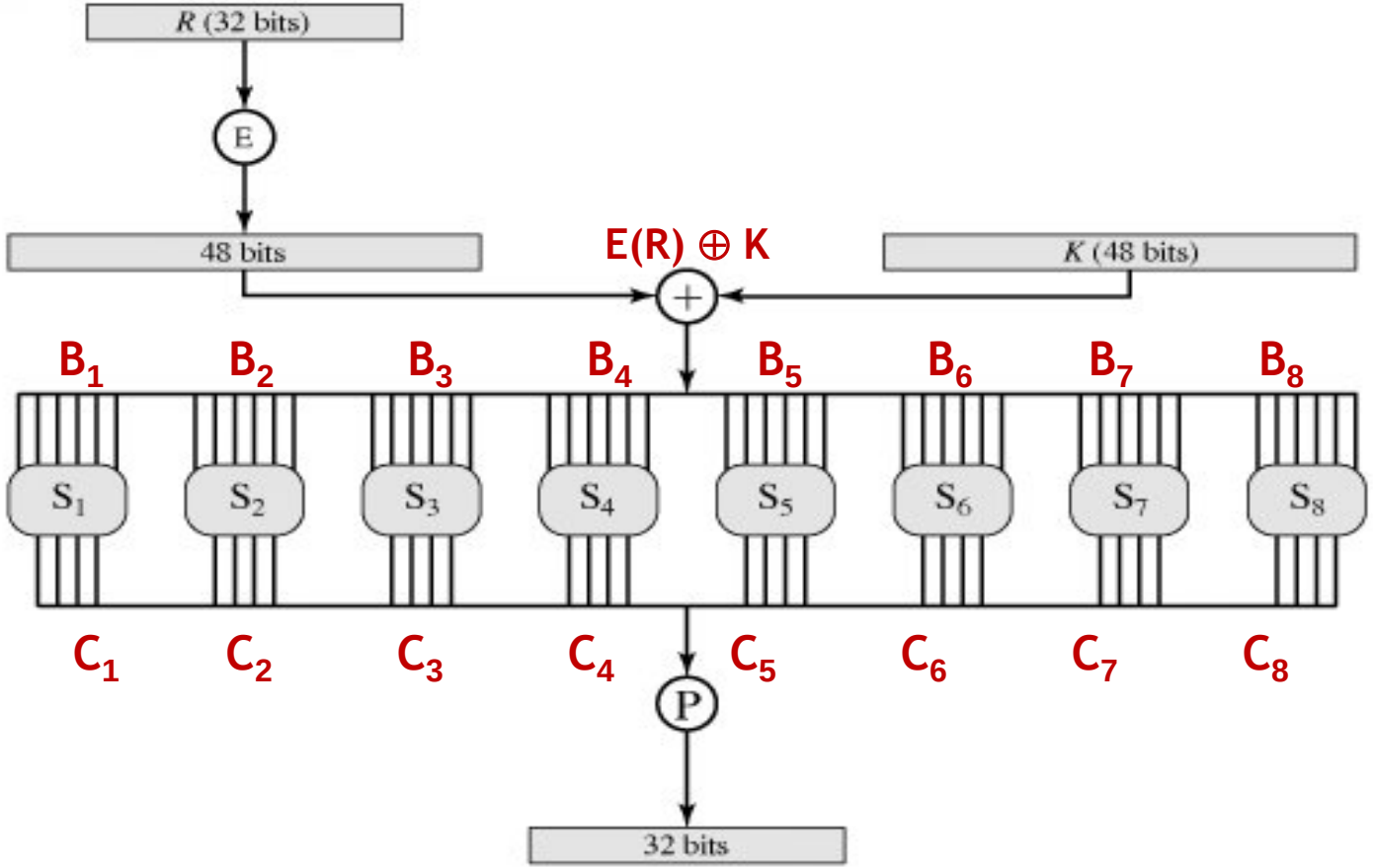
$$E(R) \oplus K = B_1B_2B_3B_4B_5B_6B_7B_8$$

Avec $Bi \in \{0,1\}^6$



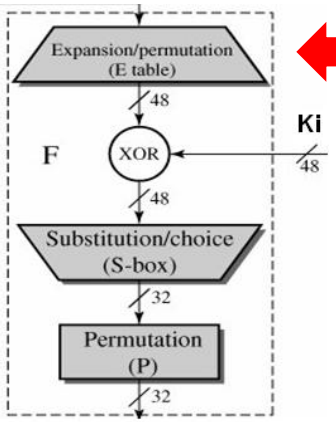
DES: Calcul médian

Détail de la fonction F:



E: Expansion , **P**: permutation

DES: Calcul médian



- **Substitution/choice (S-Box)**
- Dans l'étape suivante, on utilise 8 fonctions:

$$S_i: \{0,1\}^6 \rightarrow \{0,1\}^4$$

i: 1… 8

On les appelles S-box:

- On calcule **Ci = Si (Bi) ,**

Pour $b_1b_2b_3b_4b_5b_6$ →

b_1b_6 ➔ ligne

$b_2b_3b_4b_5$ ➔ colonne

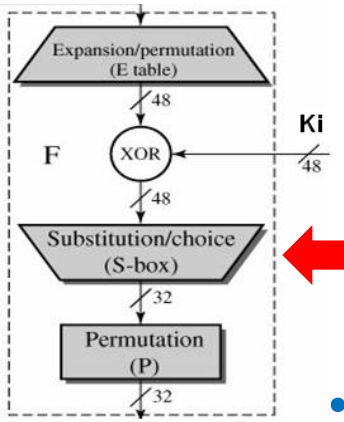
- **o**n obtient:

C= $C_1C_2C_3C_4C_5C_6C_7C_8$

C est une suite binaire de longueur 32 bits.

Lignes	Colonnes															
	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]	[11]	[12]	[13]	[14]	[15]
S_1																
[0]	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
[1]	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
[2]	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
[3]	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13
S_2																
[0]	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
[1]	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
[2]	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
[3]	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9
S_3																
[0]	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
[1]	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
[2]	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
[3]	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12
S_4																
[0]	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
[1]	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
[2]	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
[3]	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14
S_5																
[0]	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
[1]	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
[2]	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
[3]	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3
S_6																
[0]	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
[1]	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
[2]	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
[3]	4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13
S_7																
[0]	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
[1]	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
[2]	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
[3]	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12
S_8																
[0]	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
[1]	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
[2]	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
[3]	2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

DES: Calcul médian



- **Exemple:**

On veut calculer $C_1 =? S_1 (B_1)$

Etant donnée B_1= **001001**

$b_1b_2b_3b_4b_5b_6$ → **011001**

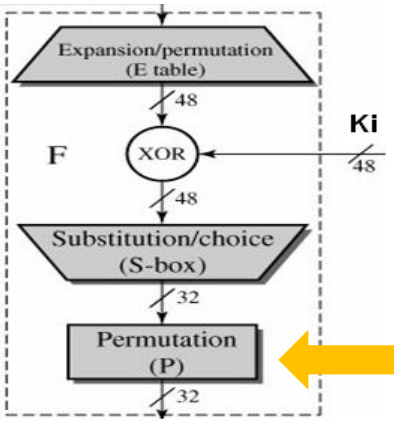
b_1b_6 → 01 ligne (1)

$b_2b_3b_4b_5$ → 1100 colonne (12)

Lignes	Colonnes															
	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]	[11]	[12]	[13]	[14]	[15]
	S_1															
[0]	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
[1]	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
[2]	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
[3]	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

- on obtient:

$C_1 = 9 = 1001$



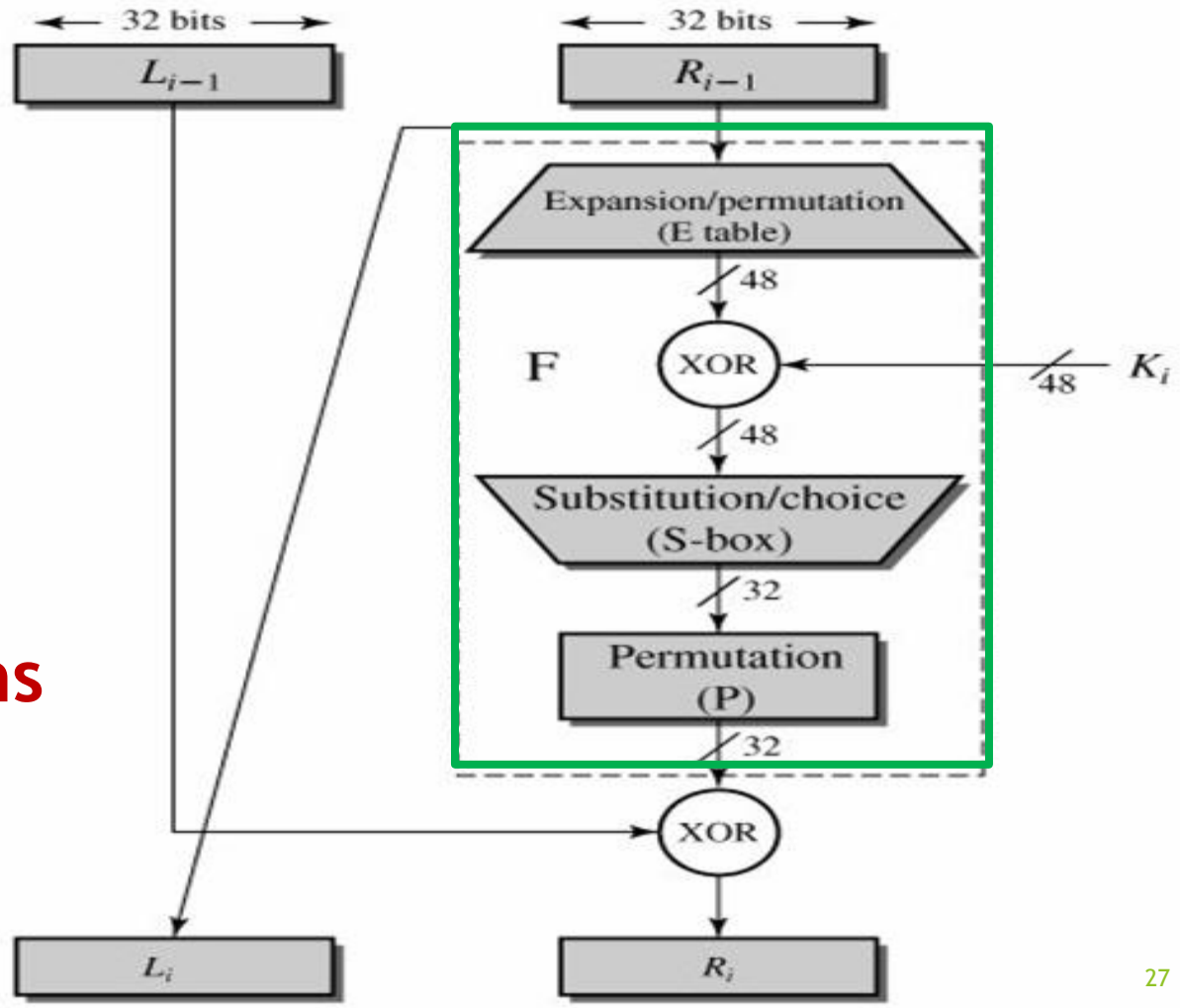
DES: Calcul médian

- **Permutation (P)**
- La permutation ***P*** est appliquée à la séquence de bits C.
- Fonction P : permutation de 32 bits

16	7	20	21	29	12	28	17
1	15	23	26	5	18	31	10
2	8	24	14	32	27	3	9
19	13	30	6	22	11	4	25

- On obtient à cet instant le cryptogramme $F(R_{i-1},K_i)$

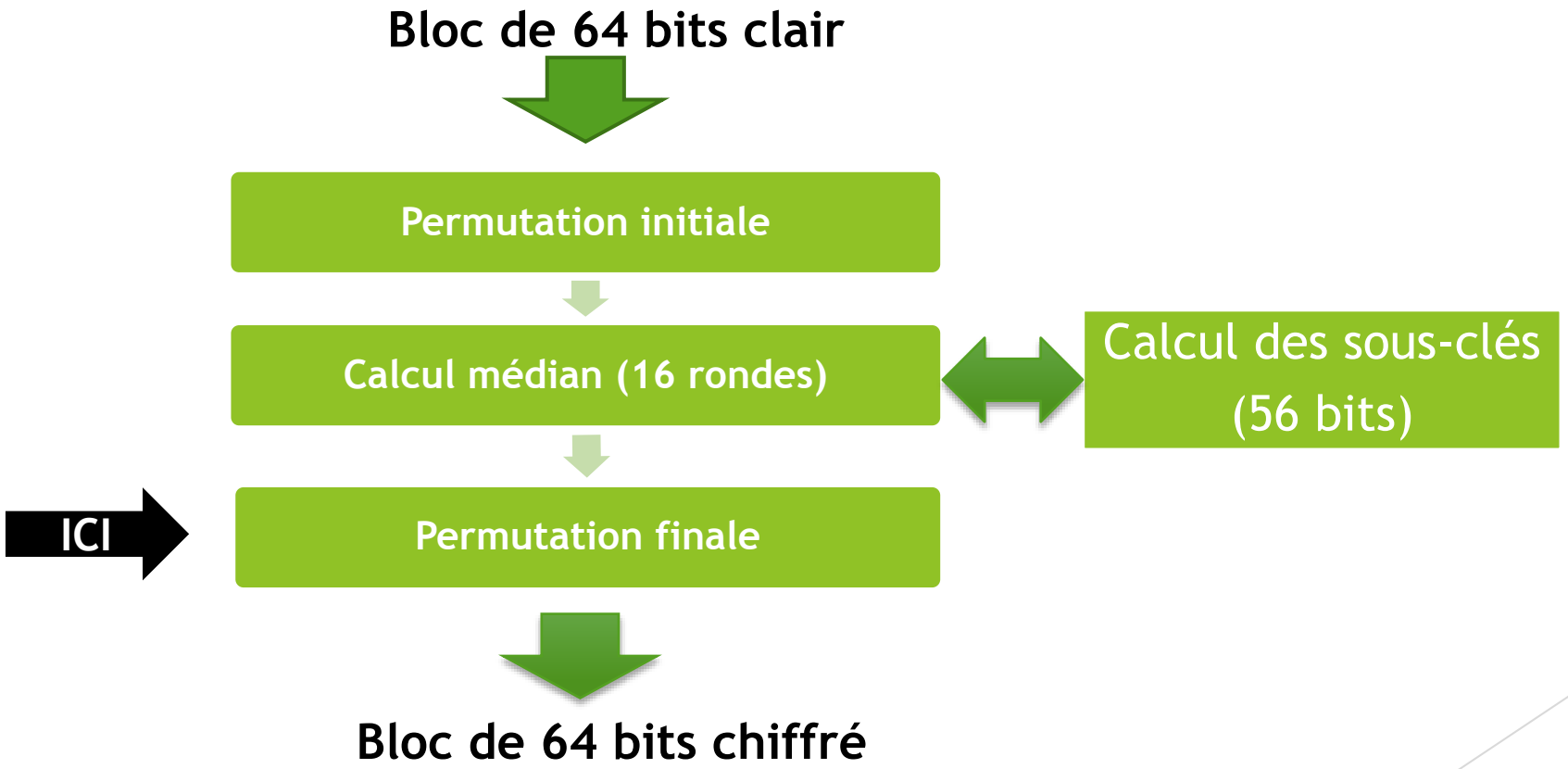
DES: Calcul médian



✓

16 itérations

DES (Data Encryption Standard)



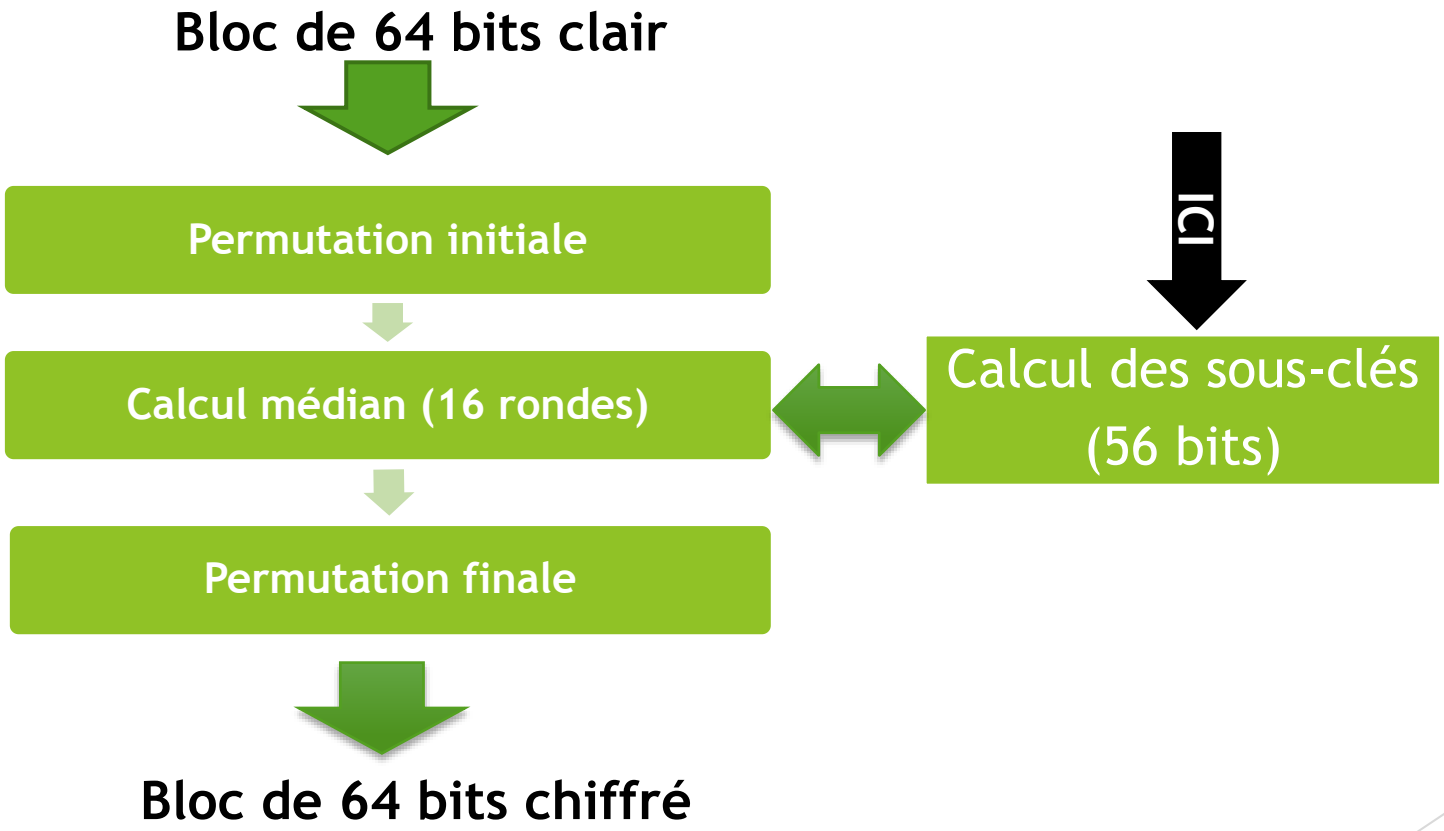
DES: Permutation finale

- Selon le tableau PI^{-1}(la Permutation Inverse de la permutation initiale):

40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25

- Le résultat obtenu est le bloc chiffré.

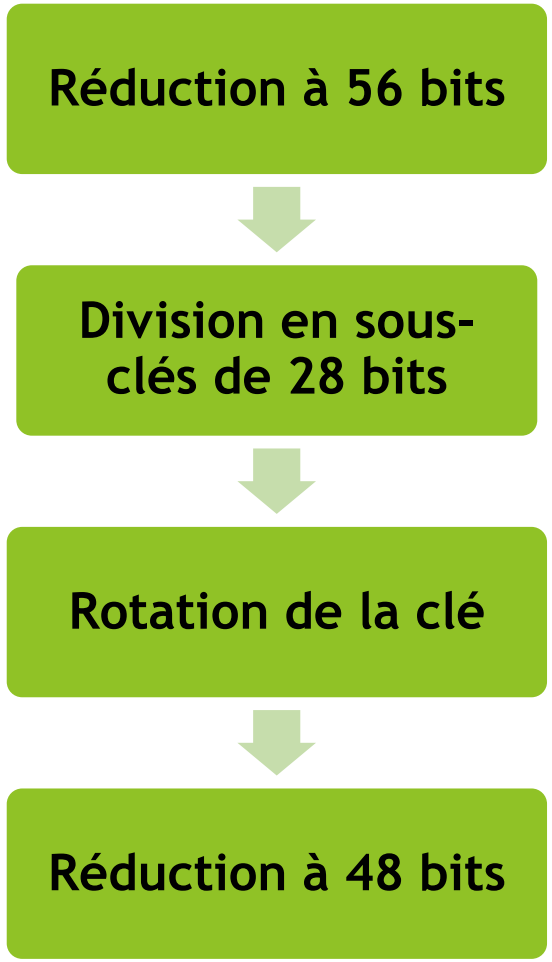
DES (Data Encryption Standard)



DES (Data Encryption Standard)

- La clé a une longueur de **64** bits, c'est-à-dire **8** caractères, mais dont seulement **56** bits sont utilisés (dans l'algorithme).
- Le nombre de clés possibles du DES est $2^{56} \approx 7.2*10^{16}$

Algorithme du calcul de la clé G(K, n)



Algorithme du calcul de la clé G(K, n)

Le calcul a lieu en 4 étapes:

1. **Réduction à 56 bits:** on utilise la fonction PC1:

57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	35	27
19	11	3	60	52	44	36
63	55	47	39	31	23	15
7	62	54	46	38	30	22
14	6	61	53	45	37	29
21	13	5	28	20	12	4

Algorithme du calcul de la clé G(K, n)

❷ **Division en sous-clés de 28 bits:** le résultat de l'étape précédente (56 bits) est divisée en deux sous-clés de 28 bits.

❸ **Rotation de la clé:** à chaque ronde, on applique la fonction rotation vers la gauche d'1 ou 2 bits pour chaque sous-clé de 28 bits selon la table suivante:

Round number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Bits rotated	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

Algorithme du calcul de la clé G(K, n)

- **Réduction:** après concaténation des deux sous-clés précédentes, on applique la fonction PC2 pour la réduire la sous- clé de 56 bits à une sous-clé de 48 bits:

14	17	11	24	1	5	3	28
15	6	21	10	23	19	12	4
26	8	16	7	27	20	13	2
41	52	31	37	47	55	30	40
51	45	33	48	44	49	39	56
34	53	46	42	50	36	29	32

Le résultat de cette réduction est la sous-clé K_i additionnée avec $E(R_{i-1})$.

DES: Déchiffrement

- On applique le même algorithme mais on inverse seulement les étapes.
- Pour la structure de Feistel, on utilise:
  - $R_{i-1} = L_i$
  - $L_{i-1} = R_i \oplus F(R_i, K_i)$

Cryptanalyse du DES

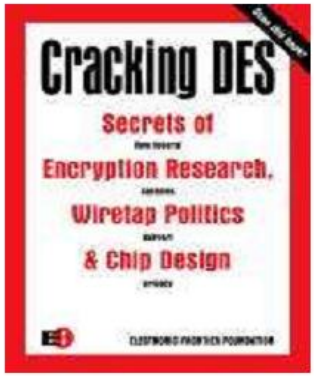
- Cryptanalyse différentielle (1991, 2^{47})
- Cryptanalyse linéaire (1994, 2^{43})
- DES Possède des clés faibles.
- Recherche exhaustive (2^{56})

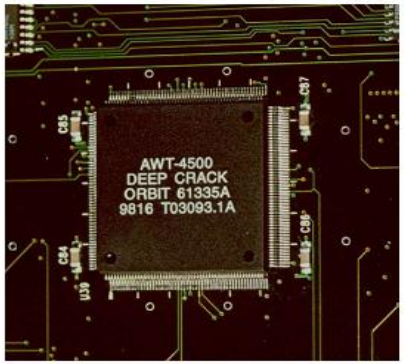
Cryptanalyse du DES: Clés faibles

- Les clés qui sont considérées comme des clés faibles dans DES sont:
  - 01010101 01010101
  - FEFEFEFE FEFEFEFE
  - E0E0E0E0 F1F1F1F1
  - 1F1F1F1F 0E0E0E0E
- **Solution:** Il faut éviter génère ces clés dans la phase de génération de clé.

Cryptanalyse du DES: Recherche exhaustive

- En 1999, distributed.net et Deep Crack ont pu casser la clé en 22 heures et 15 minutes.

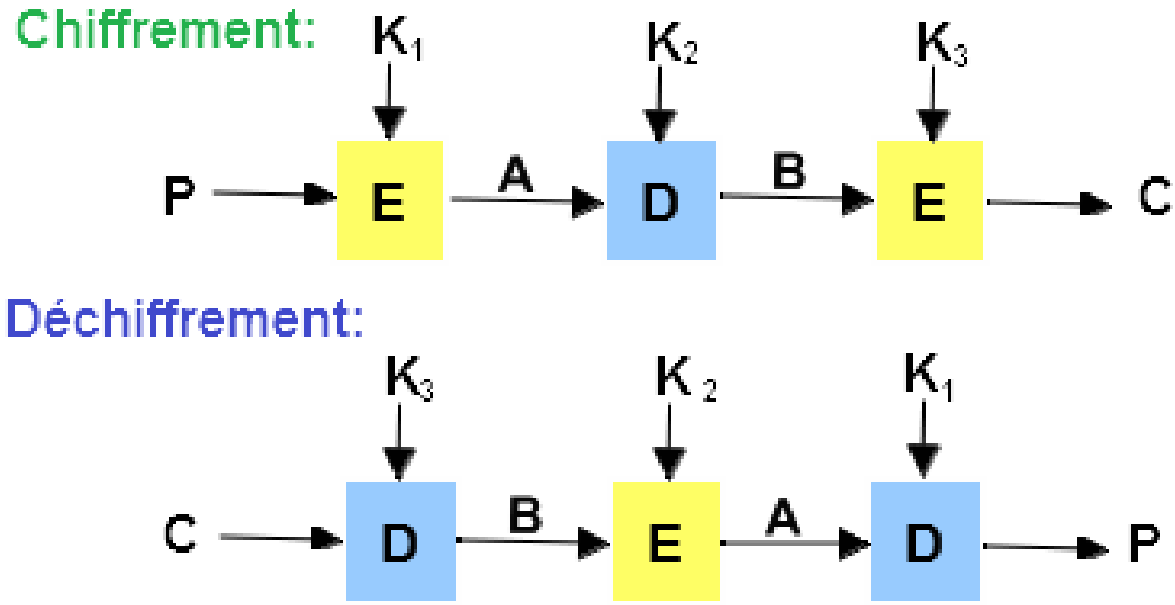




Triple DES (3DES)

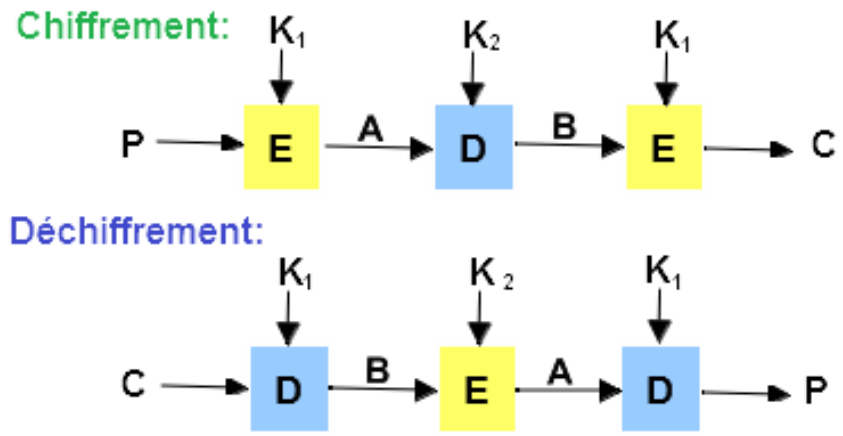
- Il est développé par W. Tuchman (IBM) en 1999.
- Il applique 3 opérations successifs du cryptosystème DES en utilisant des clés différentes.

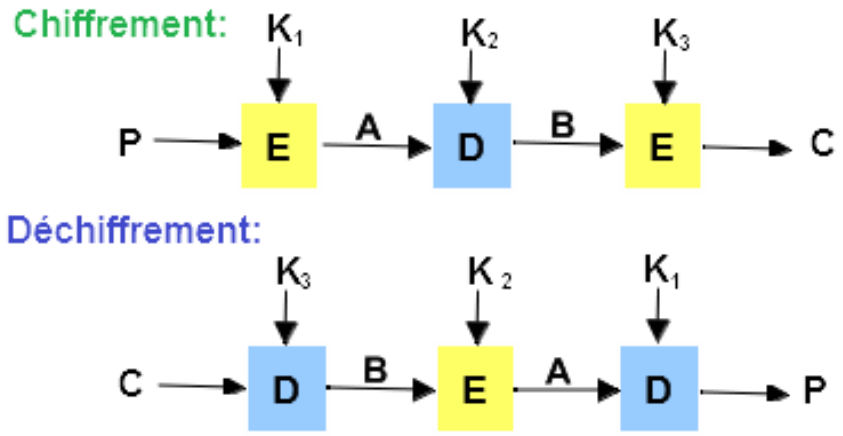
$$E(k_3, D(k_2, E(k_1, m))).$$



Triple DES (3DES)

- Il existe deux versions:
  - K_1=K_3, espace de clés 112 bits.
  - $K_1 \neq K_3$, espace de clés 168 bits.





Triple DES (3DES)

- Caractéristiques:
  - La longueur de la clé est augmentée.
  - Il résiste toutes les attaques connues (force brute, analytiques, différentielles).
  - Il est plus lent par rapport le DES parce que les opérations sont triplées.

Quiz

- Le chiffrement par flot utilise le principe de:
  - Structure de Feistel
  - Masque jetable
  - Analyse fréquentiel
  - Aucune réponse correcte
- Les tailles possibles des clés dans 3DES sont:
  - 112
  - 128
  - 168
  - 64

Références

- A. Menezes, P. van Oorschot, and S. Vanstone, Handbook of Applied Cryptography, CRC Press, 1996. (Chapter 7) www.cacr.math.uwaterloo.ca/hac
- E. Bersson, Cryptographie, Laboratoire de cryptographie. https://docplayer.fr/5870683-Cryptographie-chiffrement-symetrique.html
- R. Dumont, Cryptographie et Sécurité informatique, 2010. http://www.montefiore.ulg.ac.be/~dumont/pdf/crypto09-10.pdf
- NIST Special Publication 800-67 Revision 2, Recommendation for the Triple Data Encryption Algorithm (TDEA) Block Cipher, 2017. https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-67r2.pdf